AMENDMENTS TO THE CLAIMS

Please amend all claims as noted below:

1. (Currently amended) A computer implemented method for downloading resources, from a source to an intermediate storage facility(ies), having a finite storage capacity, the computer implemented method comprising the following computer executable acts:

determining a probability of using a resource, the probability in part determined *via*;

accepting at least one user-based factor;

accepting at least one resource-based factor;

maximizing an expected value of downloaded resources *via* utilization of the at least one user-based factor and the at least one resource-based factor;

distributing downloaded resources among a plurality of storage facilities to minimize total request-to-receive time, and

evaluating a cost of accessing resources in an unloaded condition.

2. (Previously Cancelled.)

3. (Currently amended) The computer implemented method of claim 1 further comprising determining probabilities that a user belongs to various user type classes.

4. (Currently amended) The computer implemented method of claim 3 wherein the probabilities that a user belongs to various user type classes are determined based on evidence using a Bayesian network.

5. (Currently amended) The computer implemented method of claim 3 wherein the at least one resource-based factor includes probabilities that users of the various user type classes will use the resource at least once.

6. (Currently amended) The computer implemented method of claim 1 wherein the at least one resource-based factor includes probabilities that users of various user type classes will use the resource at least once.

7. (Currently amended) The computer implemented method of claim 3, wherein the at least one resource-based factor is a probability that the resource will be used at least once and is based on a sum, over all user type classes, of a product of :

a probability that the resource is used at least once, given that an application to which the resource belongs is used at least once, by a user of the user type class;

a probability that the application to which the resource belongs is used at least once by a user of the user type class; and

a probability that the user belongs to the user type class.

8. (Currently amended) The computer implemented method of claim 1 wherein the at least one resource-based factor includes an association of each of the resources to at least one application class.

9. (Currently amended) The computer implemented method of claim 8 wherein the at least one resource-based factor includes an indication, for each of the resources, of whether the resource is a core component or an optional component of the application class with which it is associated.

10. (Currently amended) The computer implemented method of claim 1 wherein the act of maximizing an expected value of downloaded resources includes maximizing an expected value density of downloaded resources.

11. (Currently amended) The computer implemented method of claim 1 wherein the act of maximizing an expected value of downloaded resources includes minimizing an expected cost of not having a needed resource.

12. (Currently amended) The computer implemented method of claim 11 wherein the expected cost of not having a needed resource is based on one of enhancement rates of the resources and value densities of the resources.

13. (Currently amended) The computer implemented method of claim 12 wherein the enhancement rate of a resource is based on the size of the resource, a probability of that resource being used at least once, and a cost of later downloading the resource.

14. (Currently amended) The computer implemented method of claim 12 wherein the value density of a resource is based on the size of the resource and the probability that the resource will be used at least once.

15. (Currently amended) A computer ~~[executable]~~ implemented system for downloading resources, comprising the following computer executable components:

means for storing at least one user-based factor and at least one resource-based factor;

means for maximizing an expected value of downloaded resources *via* utilization of the user-based factor and the resource-based factor;

means for intelligently downloading a resource based on a probability of use to intermediate storage facilities,

means for optimizing distribution over intermediate storage facilities to minimize total request-to-receive times, and

means for evaluating a cost to retrieve resources in a non-downloaded condition.

16. (Currently amended) A computer implemented method of downloading a resource to an intermediate storage facility comprising the following computer executable acts:

accepting at least one user-based factor;

determining a probability of use for a resource by a user in a user type class;

accepting at least one resource-based factor;

maximizing an expected value of downloaded resources *via* utilization of the at least one user-based factor and the at least one resource-based factor, and

changing a storage capacity of the intermediate storage facility based on a change of the expected value.

17. (Currently amended) A computer implemented method for installing software components, each having a size, from a source to an intermediate storage facility, the method comprising, the following computer executable acts:

predicting an expected frequency of use for a software component, in part *via:*

accepting at least one user-based factor;

accepting at least one component-based factor; and

changing a storage capacity of the intermediate storage facility based on a value and cost associated therewith, and

determining a cost of accessing a resource in an unloaded condition.

18. (Currently amended) The computer implemented method of claim 17 wherein the at least one user-based factor includes probabilities that a user is member of various user type classes.

19. (Currently amended) The computer implemented method of claim 17 wherein the at least one component-based factor includes an association of each of the software components to one of a plurality of application classes.

20. (Currently amended) The computer implemented method of claim 19 wherein the at least one component-based factor further includes an indication, for each of the software components, of whether the software component is a core component or an optional component of the application class with which it is associated.

21. (Currently amended) The computer implemented method of claim 20 wherein the at least one component-based factor further includes probabilities that each of the software components will be used at least once by users of various user type classes.

22. (Currently amended) A computer implemented method for distributing resources, each having a size, among at least two storage facilities, the method comprising the following computer executable acts:

accepting at least one user-based factor;

accepting at least one resource-based factor;

accepting at least one storage facility-based factor;

accepting probabilistic relationships between user based factors and resource based factors;

minimizing total expected request to receive time *via* utilization of the user-based factor, the resource-based factor, and the storage facility-based factor; and

changing a storage space associated with the intermediate storage facility, based on the minimizing act.

23. (Currently amended) The computer implemented method of claim 22 wherein the at least one user-based factor includes probabilities that a user belongs to various user type classes.

24. (Currently amended) The computer implemented method of claim 23 further comprising determining the probabilities that a user belongs to various user type classes.

25. (Currently amended) The computer implemented method of claim 24 wherein the probabilities that a user belongs to various user type classes are determined based on evidence using a Bayesian network.

26. (Currently amended) The computer implemented method of claim 23 wherein the at least one resource-based factor includes frequencies at which users of the various user type classes will use each of the resources.

27. (Currently amended) The computer implemented method of claim 26 wherein the at least one storage facility-based factor includes an available capacity of each of the two storage facilities and a relative request-to-receive latency of each of the two storage facilities.

28. (Currently amended) The computer implemented method of claim 27 wherein the total expected latencies is a function of the frequencies at which users of the various user type classes will use each of the resources, and a difference between the relative request-to-receive latencies of the two storage facilities.

29. (Currently amended) The computer implemented method of claim 22 wherein the at least one storage facility-based factor includes an available capacity of each of the two storage facilities and a relative request-to-receive latency of each of the two storage facilities.

30. (Currently amended) The computer implemented method of claim 22 wherein the total expected latencies to request and receive resources is minimized based on value densities of the resources.

31. (Currently amended) The computer implemented method of claim 30 wherein the value densities of the resources are based on the frequency of use of the resources and a difference in request to receive latencies between the at least two storage facilities.

32. (Currently amended) A computer implemented method of [~~for~~] distributing resources, each having a size, among at least two storage facilities, each of the storage facilities having a finite available capacity, the computer implemented method comprising the following computer executable acts:

a first determining a probability of using a resource by a composite user;

a second determining, for each resource, a change in value of storing the resource on a first storage facility versus storing the resource on a second storage facility;

determining, for each resource, a change in cost of storing the resource on the first storage facility versus storing the resource on the second storage facility;

determining, for each resource, a value density in a knapsack approximation procedure based on the change in value *via* the first determining act and the second determining act; and

maximizing a total value density given a total size of resources being less than the finite available capacity of the first storage facility.

33. (Currently amended) The computer implemented method of claim 32 wherein the value of storing a resource on the first storage facility is a function of a perceived utility of such storage, per request for the resource, and a frequency of requests for the resource.

34. (Currently amended) The computer implemented method of claim 33 wherein the perceived utility of such storage, per request for the resource, is a function of a request-to-receive time delay.

35. (Currently amended) The computer implemented method of claim 34 wherein the request-to-receive time delay is a function of at least one of:

a storage device read access time,

a network speed,

a network latency, and

the size of the resource.

36. (Currently amended) The computer implemented method of claim 35 wherein the network speed is a function of a user configuration.

37. (Currently amended) The computer implemented method of claim 33 wherein the frequency of requests for the resource is a function of a user type class and a number of users belonging to the user type class.

38. (Currently amended) The computer implemented method of claim 32 wherein the cost of storing a resource on the first storage facility is a function of the resource size.

39. (Currently amended) A computer implemented system for distributing resources, each having a size, among at least two storage facilities, each of the storage facilities having a finite capacity and a request-to-receive latency, the computer implemented system comprising the following computer implemented components:

storage means for storing at least one user-based factor, at least one resource-based factor, and at least one storage facility-based factor;

means for minimizing total expected latencies to request and receive resources,

means for intelligently downloading resources; and

means for evaluating a cost to return to resources in non-downloaded condition.

40. (Currently amended) A computer implemented method of downloading a resource(s) to an intermediate storage facility comprising the following computer executable acts:

accepting at least one user-based factor;

accepting at least one resource-based factor;

accepting at least one storage facility-based factor;

accepting probabilistic relationships between the at least one-user based factor and the at least one resource based factor;

minimizing total expected latencies to request and receive resources, and

determining a cost of retuning to resources in an unloaded condition.

41. (Previously Cancelled.)

42. (Currently amended) The computer implemented method of claim 40 further comprising changing a storage capacity of the storage medium based on at least one of a change in value and cost.

43. (Currently amended) The computer implemented method of claim 42 further comprising changing the storage capacity when a ratio of value to cost is greater than one.

44. (Currently amended) The computer implemented method of claim 43 wherein the at least one user-based factor is a function of a time offline until the intermediate storage facility is reconnected with a source.

45. (Currently amended) The computer implemented method of claim 44 wherein the time offline is a probability distribution considering at least one of:

a resource context,

a user type class, and

a recent usage pattern.